A red and blue logo

Description automatically generatedWhat is QuanClick?

QuanClick is a user-configurable mouse and keyboard code generator.

It reacts to presses from a 6-button keypad and sends sequences of mouse, or keyboard, control codes over USB just like a real mouse or keyboard would do.

Because you can send sequences of codes (or macros) you can replace many mouse or keyboard actions with just one button press.

For example, you can program one button to type out your email address with just one click.

QuanClick uses a Raspberry Pi Pico microcontroller that we will refer to as a ‘**Pico**’ throughout this document.

A white box with numbers on it

Description automatically generated

QuanClick was designed and built by members of the  
**Quantum Technology Club**

For further details about the Quantum Technology Club, please visit

**www.quantumtech.club**

|  |  |
| --- | --- |
| Connect the cables to the buttons Attach the pins from one end of the multiway cable to the pin header on the keypad.  Your multiway cable may not have the identically coloured wires as in this photo - that’s not important, just that they are all different colours so you can easily identify them either end.  Just make sure all the connectors are pushed in fully.  The connection to the left (in this picture the pin with the black wire) is the common connection, the other 6 are for each of the buttons. | A white box with wires connected to it  Description automatically generated |
| Insert the Pico into the case Carefully insert the Pico into the printed slot as shown in the photo below. Insert one corner first then work the other corner in by pushing down on the edge of the board. It is a firm fit but be careful not to damage the board. | A white box with wires and wires  Description automatically generated with medium confidence |
| Connect the cables to the Pico Connect the wires to the Pico as shown in the photo matching your appropriate colours between button board and Pico.  Note the gap between connection 2 and 3.  Check that the common wire on the button board is connected to the 7th end connection pin on the Pico. |  |
| Close the lid Fit the lid onto the box by aligning the left (non-USB socket) side of the lid first. |  |
| Check board alignment Look through the hole in the case and check that the bottom of the Pico board is matched into the slot in the base of the case as shown in the photograph.  If the board is not aligned, then lift the lid and gently move the Pico in its lid slot and try again.  Once the board aligns well then click the lid down. |  |
| Check Micro-USB alignment Look through the hole in the case and check that the Micro-USB socket sits square in the aperture.  Insert the Micro-USB cable into the Pico. |  |
| Plug in the Pico into your computer Once you plug your Pico into your computer, Windows will install the ‘com port’.  Once installed, unplug the Pico. |  |
| Put the Pico into UF2 download mode Hold down the BOOT/SEL button on the Pico using a small screwdriver through the hole in the case.  Keep the button pressed and plug the Pico into your computer.  Once plugged in, release the BOOT/SEL button.  Windows will open an Explorer (file manager) window. |  |

# Adding CircuitPython to the Pico

Download the CircuitPython 9.1.4 .UF2 file from …

https://circuitpython.org/board/raspberry\_pi\_pico/

Drag and drop the downloaded file into the Pico root folder in the Explorer window that opened earlier.

# Download and install Thonny with Python

## Download from the Thonny web site

Download Thonny with Python from the Thonny web site at …

https://thonny.org

When the page loads, you will see the links for the computer you are using in the top right corner (as highlighted in the image below).

A white square with green and white logo

Description automatically generated

## Download from GitHub download for the portable version

The portable version of Thonny can be downloaded from the GitHub web site at …

https://github.com/thonny/thonny/releases/download/v4.1.6/thonny-py38-4.1.6-windows-portable.zip

Extract the contents of the zip file to C:\Users\{user\_name}\Thonny

## Install Thonny

Run Thonny.exe using the following settings …

Language: English (UK)

Initial Settings: Raspberry Pi (simple)

Once Thonny is up-and-running …

|  |  |
| --- | --- |
| Click in the Thonny menu located in the lower right corner of the window (it’s highlighted in red). |  |
| When the menu pops up, select **Configure interpreter …** | C:\Users\Steve\AppData\Local\Microsoft\Windows\INetCache\Content.Word\lower_thonny_menu.png |
| When the Thonny Options dialog box appears, click over the **Interpreter** tab (if it is not already selected) … | C:\Users\Steve\AppData\Local\Microsoft\Windows\INetCache\Content.Word\select_interpreter.png |
| … and then select the **Circuit Python (generic)** programming language interpreter.  Click over the **[OK]** button.  Thonny should detect your Pico and the ‘com port’ it is connected to and display it in the bottom right corner of the Thonny window. | C:\Users\Steve\AppData\Local\Microsoft\Windows\INetCache\Content.Word\select_interpreter.png |

Your computer is now set up, ready for you to program your Pico.

# Programming the Pico

## QuanClick source code

You can download the source code (QuanClick.py) from …

https://github.com/Havenview/QuanClick

Open the QuanClick.py python file in Thonny …

**File** > **Open** > **This computer**

… and then navigate to where you downloaded the file to and click over the **[Open]** button.

Run the script …

**Run** > **Run the script**

## Changing the macros

The code can be modified to set personal actions for any key, some examples pre-exist in the source code.

The definitions are listed in the ‘**button\_actions**’ array at line 27 in the code.

There are 3 types of actions – **keyboard**, **mouse\_move** and **mouse\_click**.

* The **keyboard** actions contain Keycode types which are encoded codes for each key and special keys like [SHIFT] and [WINDOWS].
* The **mouse\_move** actions contain an x,y co-ordinate move.
* The **mouse\_click** actions contain Mouse types like [LEFT\_BUTTON].

The Appendix contains the codes you can use.

# Appendix

## Keycodes

|  |  |  |
| --- | --- | --- |
| Keycode | Value | Function |
| A | 4 | a and A |
| ALT | 226 | Alias for LEFT\_ALT; Alt is also known as Option (Mac) |
| APPLICATION | 101 | also known as the Menu key (Windows) |
| B | 5 | b and B |
| BACKSLASH | 49 | \ and | |
| BACKSPACE | 42 | Delete backward (Backspace) |
| C | 6 | c and C |
| CAPS\_LOCK | 57 | Caps Lock |
| COMMA | 54 | , and < |
| COMMAND | 227 | Labelled as Command on Mac keyboards, with a clover glyph |
| CONTROL | 224 | Alias for LEFT\_CONTROL |
| D | 7 | d and D |
| DELETE | 76 | Delete forward |
| DOWN\_ARROW | 81 | Move the cursor down |
| E | 8 | e and E |
| EIGHT | 37 | 8 and \* |
| END | 77 | End (often moves to end of line) |
| ENTER | 40 | Enter (Return) |
| EQUALS | 46 | =` and ``+ |
| ESCAPE | 41 | Escape |
| F | 9 | f and F |
| F1 | 58 | Function key F1 |
| F10 | 67 | Function key F10 |
| F11 | 68 | Function key F11 |
| F12 | 69 | Function key F12 |
| F13 | 104 | Function key F13 (Mac) |
| F14 | 105 | Function key F14 (Mac) |
| F15 | 106 | Function key F15 (Mac) |
| F16 | 107 | Function key F16 (Mac) |
| F17 | 108 | Function key F17 (Mac) |
| F18 | 109 | Function key F18 (Mac) |
| F19 | 110 | Function key F19 (Mac) |
| F2 | 59 | Function key F2 |
| F20 | 111 | Function key F20 |
| F21 | 112 | Function key F21 |
| F22 | 113 | Function key F22 |
| F23 | 114 | Function key F23 |
| F24 | 115 | Function key F24 |
| F3 | 60 | Function key F3 |
| F4 | 61 | Function key F4 |
| F5 | 62 | Function key F5 |
| F6 | 63 | Function key F6 |
| F7 | 64 | Function key F7 |
| F8 | 65 | Function key F8 |
| F9 | 66 | Function key F9 |
| FIVE | 34 | 5 and % |
| FORWARD\_SLASH | 56 | / and ? |
| FOUR | 33 | 4 and $ |
| G | 10 | g and G |
| GRAVE\_ACCENT | 53 | ` and ~ |
| GUI | 227 | Alias for LEFT\_GUI; GUI is also known as the Windows key or Command (Mac) |
| H | 11 | h and H |
| HOME | 74 | Home (often moves to beginning of line) |
| I | 12 | i and I |
| INSERT | 73 | Insert |
| J | 13 | j and J |
| K | 14 | k and K |
| KEYPAD\_ASTERISK | 85 | Keypad \* |
| KEYPAD\_BACKSLASH | 100 | Keypad \ and | (Non-US) |
| KEYPAD\_EIGHT | 96 | Keypad 8 and Up Arrow |
| KEYPAD\_ENTER | 88 | Keypad Enter |
| KEYPAD\_EQUALS | 103 | Keypad = (Mac) |
| KEYPAD\_FIVE | 93 | Keypad 5 |
| KEYPAD\_FORWARD\_SLASH | 84 | Keypad / |
| KEYPAD\_FOUR | 92 | Keypad 4 and Left Arrow |
| KEYPAD\_MINUS | 86 | Keypad - |
| KEYPAD\_NINE | 97 | Keypad 9 and PgUp |
| KEYPAD\_NUMLOCK | 83 | Num Lock (Clear on Mac) |
| KEYPAD\_ONE | 89 | Keypad 1 and End |
| KEYPAD\_PERIOD | 99 | Keypad . and Del |
| KEYPAD\_PLUS | 87 | Keypad + |
| KEYPAD\_SEVEN | 95 | Keypad 7 and Home |
| KEYPAD\_SIX | 94 | Keypad 6 and Right Arrow |
| KEYPAD\_THREE | 91 | Keypad 3 and PgDn |
| KEYPAD\_TWO | 90 | Keypad 2 and Down Arrow |
| KEYPAD\_ZERO | 98 | Keypad 0 and Ins |
| L | 15 | l and L |
| LEFT\_ALT | 226 | Alt modifier left of the spacebar |
| LEFT\_ARROW | 80 | Move the cursor left |
| LEFT\_BRACKET | 47 | [ and { |
| LEFT\_CONTROL | 224 | Control modifier left of the spacebar |
| LEFT\_GUI | 227 | GUI modifier left of the spacebar |
| LEFT\_SHIFT | 225 | Shift modifier left of the spacebar |
| M | 16 | m and M |
| MINUS | 45 | -` and ``\_ |
| N | 17 | n and N |
| NINE | 38 | 9 and ( |
| O | 18 | o and O |
| ONE | 30 | 1 and ! |
| OPTION | 226 | Labelled as Option on some Mac keyboards |
| P | 19 | p and P |
| PAGE\_DOWN | 78 | Go forward one page |
| PAGE\_UP | 75 | Go back one page |
| PAUSE | 72 | Pause (Break) |
| PERIOD | 55 | . and > |
| POUND | 50 | # and ~ (Non-US keyboard) |
| POWER | 102 | Power (Mac) |
| PRINT\_SCREEN | 70 | Print Screen (SysRq) |
| Q | 20 | q and Q |
| QUOTE | 52 | ' and " |
| R | 21 | r and R |
| RETURN | 40 | Alias for ENTER |
| RIGHT\_ALT | 230 | Alt modifier right of the spacebar |
| RIGHT\_ARROW | 79 | Move the cursor right |
| RIGHT\_BRACKET | 48 | ] and } |
| RIGHT\_CONTROL | 228 | Control modifier right of the spacebar |
| RIGHT\_GUI | 231 | GUI modifier right of the spacebar |
| RIGHT\_SHIFT | 229 | Shift modifier right of the spacebar |
| S | 22 | s and S |
| SCROLL\_LOCK | 71 | Scroll Lock |
| SEMICOLON | 51 | ; and : |
| SEVEN | 36 | 7 and & |
| SHIFT | 225 | Alias for LEFT\_SHIFT |
| SIX | 35 | 6 and ^ |
| SPACE | 44 | Alias for SPACEBAR |
| SPACEBAR | 44 | Spacebar |
| T | 23 | t and T |
| TAB | 43 | Tab and Backtab |
| THREE | 32 | 3 and # |
| TWO | 31 | 2 and @ |
| U | 24 | u and U |
| UP\_ARROW | 82 | Move the cursor up |
| V | 25 | v and V |
| W | 26 | w and W |
| WINDOWS | 227 | Labelled with a Windows logo on Windows keyboards |
| X | 27 | x and X |
| Y | 28 | y and Y |
| Z | 29 | z and Z |
| ZERO | 39 | 0 and ) |

## Mouse codes

|  |  |  |
| --- | --- | --- |
| Mouse Code | Value | Function |
| BACK\_BUTTON | 8 | Back mouse button. |
| FORWARD\_BUTTON | 16 | Forward mouse button. |
| LEFT\_BUTTON | 1 | Left mouse button. |
| MIDDLE\_BUTTON | 4 | Middle mouse button. |
| RIGHT\_BUTTON | 2 | Right mouse button. |